Capability-based Authentication and Access Control in Internet of Things

Bighnaraj Mishra

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India

Capability-based Authentication and Access Control in Internet of Things

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering
(Specialization - Information Security)

by

Bighnaraj Mishra
(Roll: 213CS2164)

under the supervision of

Prof. Ashok Kumar Turuk

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India.

May 20, 2015

Certificate

This is to certify that the work in the thesis entitled **Capability-based Authentication and Access Control in Internet of Things** by **Bighnaraj Mishra**, bearing roll number 213CS2164, is a record of an original research work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in **Computer Science and Engineering** (Specialization - Information Security). Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Ashok Kumar Turuk
Professor

Acknowledgment

I take this opportunity to thank all those who have contributed in this journey.

Foremost, I would like to express sincere gratitude to my supervisor, Prof. Ashok Kumar Turuk for providing motivation, enthusiasm, and critical atmosphere at the workplace. His profound insights and attention to details have been true inspirations to my research. He has taught me to handle difficult situations with confidence and courage.

I thank all the professors of the department of Computer Science Engineering for the resources and environment they have provided for the successful completion of my work. The thesis would not have been successful without their support. Besides, I thank my friends and peers who have been a source of inspiration for the work.

I would like to thank my friends and seniors at NIT Rourkela for the help they have offered during the entire period of my stay.

I must acknowledge the academic resources that I got from National Institute of Technology Rourkela. I would like to thank the administrative and technical staff members of the Department who have been kind enough to advise and help in their respective roles.

Finally, I owe the heartfelt thanks to my parents and family members for their unconditional love, support, and patience, which has been a guiding force for the work I have done.

Bighnaraj Mishra

Abstract

Internet of Things (IoT) foresees the interaction and communication between different physical entities, which are constrained devices in this physical world. The entities also communicate with the Internet to provide solution for different complex problems. It goes for empowering future advances and dreams like, smart apartment, building automation, intelligent city construction, and e-health service. Secure data transmission is of prime importance in these scenarios. Standard IP-based security arrangements don't address this issue as they are not composed in view of the restrictions of obliged gadgets. Consequently, more lightweight security components are required. The entities in the domain of IoT come from different vendors. Authentication and Authorization of these entities in a network demands the exchange of identity, certificates and protocol suites. High computation power and memory is required for this transmission. We propose a framework in which the authentication, authorization and key distribution is delegated. It also integrates capability-based fine-grained access control of services. Our evaluation implements different cryptographic algorithms to manage authentication and authorization of the entities in the domain of IoT using this framework. The simulation measures the time unit taken for managing these security aspects. The framework is also tested in a hardware-based testbed and justifies that this framework might be used in most of the IoT domain.

Keywords: IoT, entity, Cryptography, AAKDS, access token, DTLS, certificate, COOJA, Arduino, ESP8266.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Internet of Things (IoT),coined by Kevin Ashton [1], infers a future world where both living and non-living physical entities are Internet connected and be able to communicate amongst themselves and with the web service applications. The entities attached with the micro-controllers and sensors represent hosts in the web. Henceforth, permitting the constrained entities of this real world to end up top notch nationals of the Internet. The IoT makes a framework that encourages the acknowledgment of future advancements and visions [2], for example, (i) smart homes, where most of the things can be controlled remotely, e.g. aerating and cooling, doors, windows, entryways and apparatuses; (ii) Smart urban areas, which take into consideration a more proficient administration of the city, e.g. administration of road lights, element lighting taking into account current movement stream, identifying and observing of locations of contamination/commotion/temperature of the places; (iii) Smart electricity supply system and smart meter, which in light of the customers' conduct would enhance the productivity and manageability of the generation and dissemination of power; (iv) Smart Health, where health checkup equipments are interconnected to give the medical facilities at home.

Physical entities, connected with constrained devices are commonly implanted frameworks and asset obliged as to power, calculation, and memory. Such obliged gadgets are connected and access the services from the Internet, which is untrusted. This requires some sort of security features. The recent security solutions like TLS [3] and IPsec [4] are IP-based, but since the communication cost is very high and they need lavish processing for handshaking, they are not intended for constrained devices. Henceforth, existing IP-based security standards can't be utilized effectively directly.

This demands the interest of security solutions that can be applied reliably to

the constrained devices. Security solutions ought to be standard agreeable with a specific final goal to empower the acknowledgment of the IoT vision and to encourage interoperability. In the present scenario, symmetric key based security protocols like Pre-Shared Keys (PSKs) can be used to provide host to host security services in WSNs. PSK is prominent in WSN because of it's low resource requirements for computation and verification. This methodology is adequate for segregated and neighborhood situations, where a domain administrator deals with the key dispersion. In any case, the key administration in symmetric key-based arrangements are unreasonable and not adaptable, particularly when the entities are from distinct domains as in case of IoT. The reason behind this is that a single key that is used by every host needs to be preoccupied and this key has to be predeployed.

One of the commanding security solution on the Internet to provide end to end (E2E) security is the Public Key Cryptography (PKC). It settles the key administration problems and permits message verification without a preshared key on each end. Besides, the certificate based approach used in PKC, provides extra administrations to improve the vision of IoT, where symmetric key-based solutions miss the mark. Datagram Transport Layer Security (DTLS) [5] is an IP-based security convention that accompanies adaptability in the cryptographic configuration that takes into consideration both PSK and PKC-based verification. So an interesting issue is to measure the overheads of both plans and present a thorough comparison.

1.1 Motivation

With the large use of internet and automated devices, the demand of application of IOT increases day by day. Initially RFID was considered as the only method of communication, although it also may include other sensor technologies, wireless technologies or QR codes. Now IP-based protocols and technologies are integrated with the availability of the advantages of IPV6.

The service providing entity is not only accessed by the owner, but is now connected to surrounding entities and databases. Many objects act in unison, to provide ambient intelligence. Also, they can be accessed and controlled remotely by legitimate servers other than the owner.

Considering this scenario, proper authentication and access control techniques need to be applied so that only ethic users can access and communicate with the entities.

1.2 Objectives

In this thesis, we look for an efficient standards-based security framework for IoT. The key objectives of this thesis are:

- The review of the different authentication schemes used by constrained networks;
- Design of a capability-based authentication and authorization framework for IOT to overcome the overheads of mutual certificate sharing authentication scheme;
- The framework facilitates the security services;
- The authorization should be fine-grained;
- The computation power constraint and memory availability constraint are taken care of.

1.3 Thesis Organization

We divide our work as follows. Chapter 2 explores about constrained entities, cryptographic requirements and DTLS protocol suite. Chapter 3 formulates the problem statement and propose a delegation-based framework to enable security services for constrained devices in IoT. In Chapter 4 we implement and evaluate the framework using COOJA simulator [6] and analyze the time unit taken for different events of our framework. Chapter 5 analyzes the memory space requirement of our framework, through hardware emulation, using Arduino micro-controller. Finally, we conclude this thesis in Chapter 6.

Chapter 2

Security Aspects in IoT

The essential literature survey for this thesis is presented here. In the first section, we talk about the characteristics of the constrained devices and the network in which they function. Then, a brief overview on relevant cryptographic requirements are highlighted. We talk about the Datagram Transport Layer Security (DTLS) protocol at last.

2.1 Constrained Entities and the Network Environment

Constrained devices are furnished with restricted memory, CPU and power facility. The network environment consisting of these devices makes the physical world smart through sensors, communication and activating usefulness into intense devices. In this section, we quickly talk about the qualities of such compelled substances concerning IOT and the systems they work in.

The constrained entities are attached with Micro-Controller Units (MCU),or tiny sensors and motors to be used as standalone devices. They come with low computation-power CPUs, and small size of storage for code and information. Besides, low power consumption becomes essential since most of them may be battery-controlled. Such constrained devices mostly impart remote, while outskirt switches and Gateways (GWs) that associate one network like Internet with a Wireless Sensor Network (WSN) , may have wired connection [7]. IEEE 802.15.4 [8] standard is a mostly used data link layer technology for WSNs.

Some other low-power technologies like Low-Power IEEE 802.11 [9] and Bluetooth Low Energy (BLE) [10], are already in function. Be that as it may, these innovations

are at present being created and are not yet generally conveyed in WSNs.

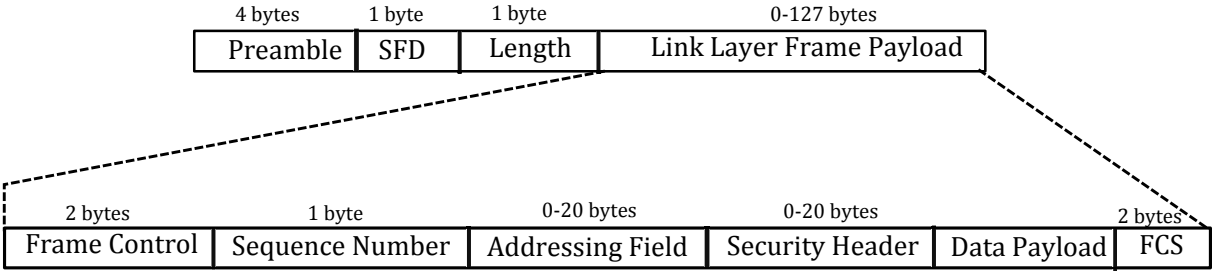


Figure 2.1: Frame Structure in IEEE 802.15.4.

Constrained devices commonly work in low-control IP systems, which are because of the confined way of such implanted gadgets with restricted memory, power and computational facility. The rare resources leads to smaller packet queueing conceivable outcomes in a compelled hub, which part of the way begins the lossy way of Low power and Lossy Networks (LLNs). The frame arrangement in the MAC and physical layers in IEEE 802.15.4 is shwon in Figure 2.1.

2.2 IP Connectivity

Internet of Things (IoT) has a vision of conveying the network for things sake to the Internet. Constrained entities become empowered through this integration prerequisite. Moreover, because of the far reaching ability of IP network, IP-empowered systems are more proficient with respect to upgrade. The utilization of a typical convention stack, for example, IP, takes into account interoperability of heterogeneous devices from distinctive producers. At that point, all devices utilize a very much standardized protocol suite. Besides, a standard protocol suite makes the devices independent of link layer.

Constrained systems are getting to be IP-empowered systems and subsequently moving far from segregated WSNs towards interoperable and web-enabled systems. For this an IP supporting layer is needed which helps in modifying IP packets for the routing of the packets in constrained network, for example, IEEE 802.15.4-based systems. IPv6 over Lowpower Wireless Personal Area Network (6LoWPAN) [11, 12] is such a layer for IEEE 802.15.4-based systems. It resides between the Network layer and the Data Link Layer. For this thesis 6LoWPAN is pertinent, since its usefulness influences the integration and henceforth the secure communication.

2.3 Requirement of Cryptography

Cryptography is basic to give security in an IOT feel. Hence, a brief diagram about security objectives and real dangers in system security are examined here. Subsequently, the idea and application of symmetric cryptography is visualized. We cover Public Key Cryptography (PKC) and structure of X.509 certificates [13] towards the end.

2.3.1 Security and Threats

Cryptographic primitives are used to provide the prime security goals for exchanging messages and to protect the network itself. These goals are: (i) *integrity*, the original message remains intact, (ii) *confidentiality*, only the authorized entities can access the data, (iii) *authenticity*, the entity genuineness is verified, and (iv) *availability*, the system provides service continuously to the legitimate entities.

Assault methods are critical to comprehend the reason of security components in communication protocols [14]. The accompanying attacks are essential concern to secure Internet of Things (IoT), : (i) *Eavesdropping* is the procedure of unauthentic accessing of data stream during a communication. (ii) *Impersonation* is the point at which a malevolent substance puts on a show to be as an authentic element (iii) The *MITM* (Man In The Middle) attack happens when a noxious element intrudes on the correspondence of two genuine elements and is equipped for deferring, altering or dropping messages. (iv) The *DoS* attack focuses on the accessibility of a framework that offers administrations. This is attained to by thoroughly getting to assets at the casualty so that the offered administrations get to be distracted to genuine elements. For constrained devices where existing assets are as of now rare, this is very critical.

2.3.2 Symmetric and Asymmetric Cryptography

In Symmetric cryptography both the end-points share a single key. The encryption and decryption operations of the cryptographic algorithms are performed using this secret key. The Symmetric-key cryptography algorithms are broadly divided into two categories: block ciphers and stream ciphers. Because of the use of the constrained entities in the environment of IOT, light-weight cryptographic algorithms are generally used. Some of the prominent light-weight cryptographic algorithms are AES, PRESENT, HUMMINGBIRD, etc.

One major problem in the symmetric cryptography is the the single key has to be shared between the endpoints. Use of a central key distribution point can solve this issue. But a secure E2E connection has to be established between the entities and the key distribution center. Public Key Cryptography (PKC) or Asymmetric cryptography is the alternate way of achieving security goals. It uses two different keys for encryption and decryption. The keys are derived based on hard problems that require a lot of computational power to solve like: prime factorization problem, discrete logarithm problem and a high numeric power calculation problem. The prominently used public-key cryptosystems are RSA, ECC, ECDSA, ECDH, etc.

2.3.3 X.509 Certificates

X.509 certificates are standard certificates prominently used to provide authentication. The structure of X.509 certificates is discussed here.

- *Version number*: defines the version of X.509.
- *Serial Number*: defines a number assigned to each certificate.
- *Signature algorithm ID*: defines the algorithm used to sign the certificate.
- *Issuer name*: defines the certification authority that issued the certificate.
- *Validity period*: defines the earliest time and the latest time the certificate is valid.
- *Subject name*: defines the entity to which the public key belongs.
- *Subject public key*: defines the owners public key.
- *Issuer unique identifier* (optional): using unique value here, the two issuer can have a same Issuer name field.
- *Subject unique identifier* (optional): using unique value here, the two issuer can have a same Subject name field.
- *Extensions* (optional): more private information can be added to the certificate using this.
- *Signature*: used for authentication of the certificate.

2.3.4 Implicit Certificate

Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV) needs less public-key-based operations [15]. It's size is small too. So it is more efficient and suitable for constrained network as compared to traditional X.509 certificates.

A X.509 certificate is made out of the three fundamental components: Public Key PK_X of entity X, data $DATA_X$ of that entity, and the signature $SIGN_{CA}$ of the past two components marked by a CA. Conversely, an implicit certificate just conveys the data component $DATA_X$. This helps in reduce the certificate size, which is very short like the measure of the utilized elliptic curve public key, for occurrence 256 bit. The public key PK_X and $SIGN_{CA}$ are superimposed to $DATA_X$. The public key PK_X is recreated from $DATA_X$ given public key of the CA. Moreover, it is expected that the utilized elliptic curve is known to all parties. In ECQV, the coupling of the public key to the proprietor is exhibited by demonstrating ownership of the private key relating to the recreated public key. This certainly tying requires further message trades and the utilization of the private key.

2.4 IP-based Security Protocol and DTLS

(TLS) [3] is a noticeable IP-based security convention broadly utilized as a part of the Internet. It makes a straightforward association arranged secured channel that avoids security attacks, for example, altering the message, eavesdropping, or message imitation. The broadly utilized web protocol HTTP uses TLS protocol as the security convention. The reliable Transmission Control Protocol (TCP) underlies this HTTP during communication. Because of the small overhead and simple structure, UDP is prominently used in the IP-empowered constrained networks, for the on-interest communication example of such systems. Because of all these reasons, the standardization community like Internet Engineering Task Force (IETF) prefers to use the Datagram Transport Layer Security (DTLS) [5] as the important transport layer security protocol.

The Datagram Transport Layer Security (DTLS) protocol structure consists of the initial flights between the two entities to authenticate each other and key agreement. Then the data is transmitted as the secure channel is established. The detail structure of DTLS is shown below.

It consists of 8 flights as shown in Figure 2.2:

- *Flight1*: Client sends a hello message and waits for the acknowledgment.

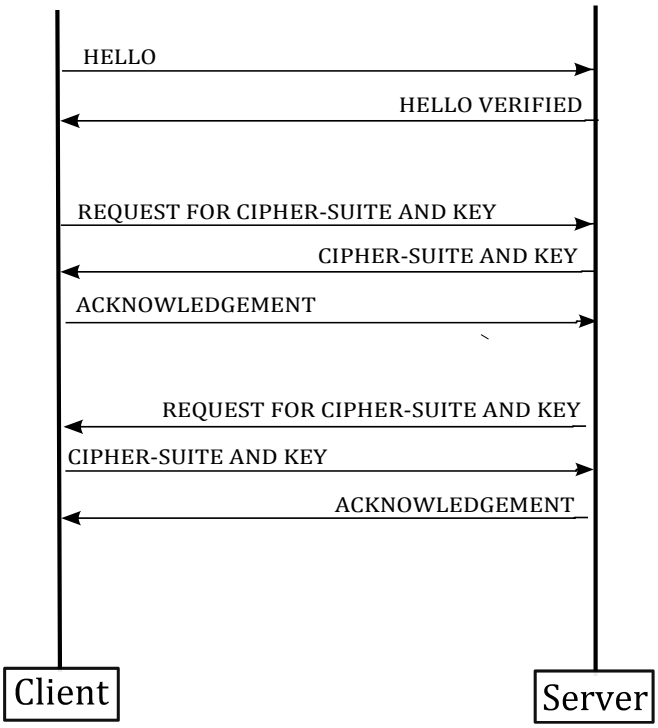


Figure 2.2: DTLS handshaking between Client and Server.

- *Flight2*: Server sends the acknowledgment.
- *Flight3*: Client requests for the certificate and cipher suite from Server.
- *Flight4*: Server sends its cipher suite and certificates.
- *Flight5*: Client sends an acknowledgment after receiving.
- *Flight6*: Server sends a request for the client certificates and cipher suite.
- *Flight7*: Client sends its cipher suit and certificates.
- *Flight8*: Server sends the acknowledgment.

2.5 COOJA Simulator and Contiki OS

Contiki [16] is a widely used open-source Operating System (OS) in micro controllers. Contiki is suitable for the Internet of Things (IoT) network because it implements with IP the full IPv6 protocol suite. It comes with hardware-specific drivers loadable modules, which make it highly portable.

Commonly, Contiki processes keep running in the agreeable setting. In any case, it use preemptive setting to keep running interrupts and real-time timers. In addition, it has a multi-threading library with optional preempting. It has drivers which facilitate hardware access. So hardware-specific drivers are essential. The idea of drivers takes into consideration an encapsulated hardware access through drivers. For instance, the counters and clocks present in the hardware are used by the the current timer library in Contiki , which supplies different clock functinalities and diverse scaled clocks.

Contiki is highly portable and it has small code size, since it is written in C. Moreover, Contiki comprises of modules that can be improved by requirements. This helps in designing a modified OS by using the flexible modules already present. For example the TCP and UDP protocol suites can be altered depending upon the need of the application development. The simulation tool named Cooja is another highlight of Contiki OS, which helps in the procedure of development and debugging [6].

Chapter 3

Authorization and Access Control in IoT

We have thoroughly made a survey on the security aspects of IoT in the last chapter . We identified 4 major research directions for secure communication in the Internet of Things (IoT). They are: (i) *protocol-based extensions and optimizations*, (ii) *centralized approaches*, (iii) *solutions that require special purpose hardware modules* and (iv) *alternative delegation architectures*. In this thesis, we concentrate on the delegation architecture used for Authentication and Access Control.

3.1 Centralized approach and it's Disadvantages

Secure M2M communication methodologies using symmetric-key are efficient and suitable for constrained networks. On the other hand, a typical issue therefore is the versatility of the key administration. Two entities must share a single key for communication. So an entity must be preoccupied with all the keys with whom it wants to communicate, before the actual deployment. To handle this issue scalable and advanced key distribution techniques are introduced. Here, a key distributor is used which is nothing but a central server which performs the task of key agreement. The constrained entities are preoccupied with a secret key for safe communication with the central server. This is applicable to small domains and it requires high trust on the key distributor. However, establishing trust between key distributors of distinct domains is challenging, when two entities present in two different domains want to communicate. This requires additional non trivial infrastructure between key distributors, like Public Key Infrastructure (PKI). A centralized approach for

securing unicast and multicast communication in constrained networks is presented by Perrig et. al. called SPINS [17]. It is composed of two security protocols: the Micro Timed Efficient Stream Loss-tolerant Authentication (TESLA) and the Secure Network Encryption Protocol (SNEP). TESLA provides authenticated broadcast for constrained environments where as SNEP provides integrity and data confidentiality of unicast communication. In the bootstrapping phase, each constrained device acquires a master secret from the domain manager which could be the Gateway (GW) or sink node in a WSN scenario. A Pseudorandom Function (PRF) is used between entities to derive the Encryption and MAC keys from this master secret.

A polynomial-based scheme is proposed by Garcia-Morchon et. al. [18] to provide trusted M2M communication in IoT which is an alternative to public-key-based primitives in DTLS. The key distribution process in sensor networks is simplified here. Here every entity E is assigned with a polynomial share $P(E; b)$ derived from a secret symmetric polynomial $P(a; b)$. So only the nodes having a polynomial share, can have a secret key $P(E1; E2)$ used for communication.

3.2 Delegation-based Architectures

Delegation-based methodologies acquaint solutions to delegate computationally escalated tasks to more powerful devices. For example, public-key-based operations used in session setup is generally delegated. Server-based Certificate Validation Protocol (SCVP) [19] is one of such delegation methods. Here a trusted server receives the delegation of complex task of certificate path construction or certificate validation. Clients needn't bother with the particular usage for authentication acceptance and could thus have a disentangled rationale. This, on the other hand, needs that the SCVP server ought to be trusted very much like a neighborhood programming.

If there should be an occurrence of untrusted SCVP servers, the customer can appoint less basic errands, for example, getting disavowal data by method for CRLs. A MAC or a digital signature is used by SCVP to protect the integrity of the inquiries and responses. A key agreement approach like Diffie-Hellman (DH) is used to infer the key used to make the MAC. Thus, this implies that clients have to bolster and perform extravagant public-key-based operations. Additionally, this methodology further builds overhead inside constrained networks for the handshaking operations. Because the handshake communication needs to transmit chain of certificates, which causes most noteworthy transmission overhead since the length of certificates is very

large. Moreover, extra protocol functionality related to SCVP is requierd for the constrained devices.

Bonetto et. al. [20] introduce another delegation approach concern to IoT. According to the proposal, the public-key-based operations are delegated to an all the more efficient device like the Gateway (GW). They proposed the strategy, where the GW professes to be the end point and intercepts the Internet Key Exchange (IKE) session establishment process. The constrained devices are handed over the session key after computation of the same by the GW. Starting here on, this session key is used by the entities to secure the communication.

The GW is considered as highly trusted in this technique. At this point, the GW possessing the session key, as an on-way element, can access the plain text in communication. Thus, it has the unnoticed capability of altering messages. This breaks the node to node security, which may be satisfactory in case of some application. For case, in mechanical industries the GW is profoundly trusted and this scheme can be appreciably used. A Body Area Network (BAN) is another application scenario where high trust level is given to the individual cell phone serves as the GW for highly constrained devices. In any case, always a trusted GW is generally not available in IoT vision. Case in point, in the smart urban area scenario, the network contains constrained devices of distinctive makers. These devices don't essentially impart a solid trust connection to the GW which could be from an alternate administration supplier.

3.3 IoT Scenario and Challenges

This segment portrays our objective system environment and clarifies the difficulties in empowering End-to-End (E2E) security for the Internet of Things (IoT). In the wake of contemplating the related work and getting to be acquainted with current methodologies, we can now call attention to their shortcomings and deficiencies in empowering productive secure E2E correspondence. Afterwords we figure out the objective of this thesis.

3.3.1 IoT Network

IoT network comprises of heterogeneous entities in the scope of profoundly compelled devices to capable devices, for example, the Cloud, where the backend administration can dwell. An obliged system is interconnected through a Gateway (GW) with

unconstrained systems, for example, the Internet.

Numerous network characteristics, for example, idleness and strength of the routing, are critically affected by the network topology of a constrained network environment. The nodes can shape a single-hop network like in the associated star topology, or structure a multi-hop network depending upon the distinctive conceivable network breadths. Notwithstanding, the network topology is extremely application-driven. Case in point, since passages and streets are a major aspect of the smart urban [21] the line topology is important and for sensor deployments [22] the tree topology is commonly used. Figure 3.1 shows the two noteworthy communication designs we find in our objective network scenario. First is the *inter-domain communication* between a constraint entity and a host in the Internet through a GW that empowers the inter-connectivity of two domains. The other one is the *local communication* inside the space of a constrained network.

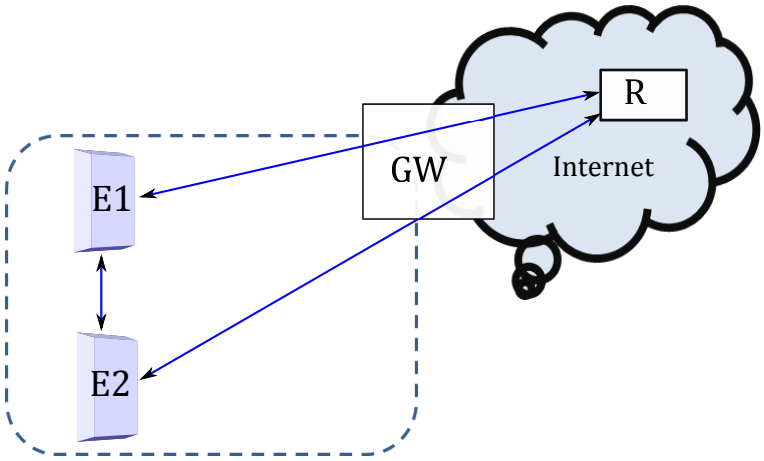


Figure 3.1: A network with constrained entities(Ei) connected with a Remote Sever(R) through a gateway GW.

3.3.2 Challenges for Secure IoT

Enabling a safe Internet of Things (IoT) needs adapting to technical and specialized scenarios [23]. In the accompanying, we quickly talk about some of the challenges needs to be considered.

- *Tight resource-constraints*: The most essential difficulties that ought to be addressed in any security arrangement regarding IoT is the little availability of the resources of constrained devices, for example, memory, CPU, data

transfer capacity and power. The small size of memory may not give sufficient space for holding data and code to implement different security approaches. The constrained CPU needs extra time for expensive encryption, decryption and certificate validation process operations. This eventually influences the responsiveness of the entity, since amid these operations the entity cannot tail its actual function. Also, this specifically influences the life time and the energy utilization of the entity.

- *Heterogeneous resources*: The existence of the heterogeneous devices is one of the unavoidable characteristics concerned to IoT. The distinctive categories of entities and the heterogeneity has to be taken care of while making the security arrangements for the extensive variety of device sorts.

- *Bootstrapping*: The life cycle of a constrained entity begins with the deployment and the first run through bootstrapping. Afterwords, a long haul operational mode takes after where the application is running and performing predefined undertakings. Amid a long lifetime, incidental upkeep/reconfigurations and bootstrapping may happen. A security arrangement ought to additionally consider the bootstrapping of the security parameters.

- *Scalability*: Adding nodes incrementally to a current network is a regular process. Moreover, particularly in an IoT environment, a node ought to have the capacity to make a protected E2E association with a variable number of companions. Also, it ought to be conceivable to subjectively include and uproot secure associations during its execution.

- *Trust Level*: The current trust level in a system is fluctuated and extremely application subordinate. Case in point, in many organizations a GW exists that interfaces multiple domains, the Internet and the constrained networks. GWs are commonly not battery-controlled and are somewhat mains-fueled and all the more capable as opposed to constrained entities .

Moreover, due to the heterogeneity of organizations, all the more capable entities may be existing in the same network. Such intense nodes and GWs are contender to deal with costly operations in the interest of constrained devices and dis-burden their restricted assets. Then again, with regards to designating cryptographic operations to all the more effective nodes, a certain level of trust is needed. Such a solid level of trust may be available in a Body Area Network (BAN), where the individual cell

phone serves as the GW for the body sensors. Too in an undertaking organization, the GW could appreciate a solid trust level. Then again, there are too other application situations where the GW can just in part be trusted. Case in point, in the city automation, constrained devices of diverse assembling/administration suppliers utilize the administrations of a GW which may be from an alternate administration supplier and consequently, the GW can't be trusted with touchy information. Consequently, the expected trust level in a network is also an imperative variable for planning suitable security arrangements. Case in point, selecting an off-way entity for assigning crypto operations has the favorable position that the entity does not have understood access to the delicate information activity.

3.4 Problem Statement

The authentication is generally established using DTLS handshaking protocol. If there should arise an occurrence of authentication using mutual certficate sharing, DTLS actuates overheads in manifestation of memory portion and communication. The vast handshake messages bring the communication overhead. For processing these large messages sufficient buffers are required. Again more overhead occurs to check and approve the authentication of the certificates. The previously stated contemplations need a definite examination of the overheads, which we handle in our work. This permits us to have the capacity to devise ways to deal with lessen these overheads.

Secure IoT needs to achieve the goals of security that is confidentiality, integrity, and authenticity to provide secure M2M communication between entities and Internet. Current methods use pre-shared keys on both ends and certficate-based schemes which are infeasible for constrained entities. We argue that PKI included with IP-based authentication can be deployed.

In this scenario, we concentrate on DTLS as the mean to attain secure communication. To measure the resource requirements and overhead, we need to implement a secure IoT network. This ought to be as lightweight as could be expected under the circumstances, to fit the accessible resources of constrained entities. While growing such an execution, overheads already existing can be recognized and answers for lessen them contrived. Delegation approach is used to perform the expensive PKC operations. This would permit using the advantages of PKC, for example, key agreement without earlier learning, with an extensive variety of devices. All the more critically, certificate-based authentication of the server can be done which is generally

performed for web services. In any case, the overwhelming PKC operations could be appointed to an all the more intense off-way device that satisfies the trust level.

At the same time we integrate *capability-based security approach* [24] to manage access control in IoT. A capability is an access token that is used for authentication and access control. It refers to a value that uniquely identifies an entity with the set of access rights approved to that entity. Capability-based authorization has certain advantages like: it helps in delegation, it supports granularity in access control and helps in revoking authorization. These advantages make this approach important for a fine-grain based access control environment.

3.5 Proposed Framework

The DTLS handshaking is delegated to a powerful server called Authentication, Authorization and key distribution server (AAKDS) as shown in Figure 3.2.

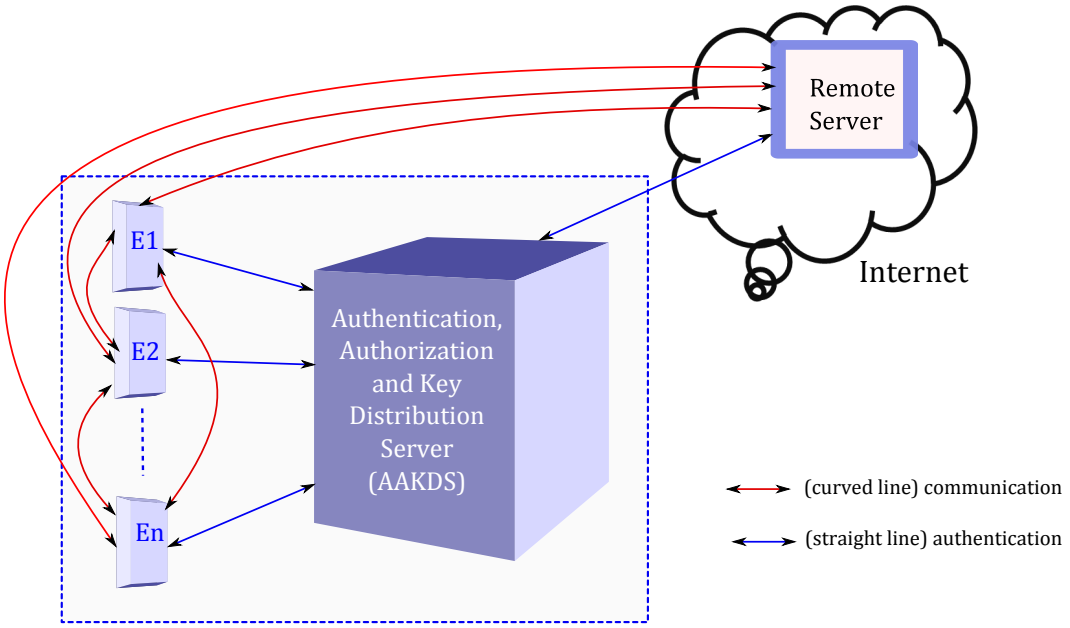


Figure 3.2: Proposed Framework.

The model consists of 4 events: Entity Registration, Remote Server Registration, Authentication of entities, communication.

3.5.1 Registration of Entities

An entity comes with a predefined key and a predefined protocol suite. The entity must be registered in the home network before the actual communication takes place. The steps followed for registration is shown in Figure 3.3.

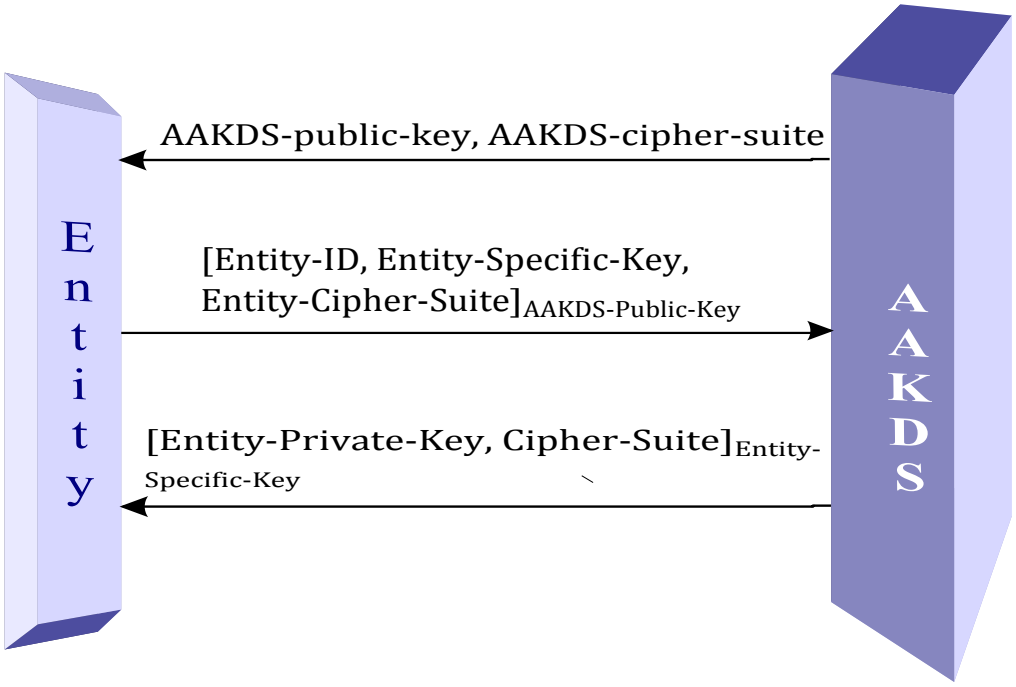


Figure 3.3: Entity Registration Process.

Procedure 1 : Entity Registration.

1: The AAKDS publishes its public key.

2: The Entity sends its credentials (Entity-specific-key and its Entity-specific-ID) and its cipher suit to AAKDS, encrypted with the AAKDS-public-key.

3: AAKDS stores the credentials of the Entity in an encrypted form and sends a private-key to the entity encrypted with the Entity-specific-key.

3.5.2 Registration of Remote Server

The Remote Server is registered using the DTLS handshaking as shown in Figure 3.4.

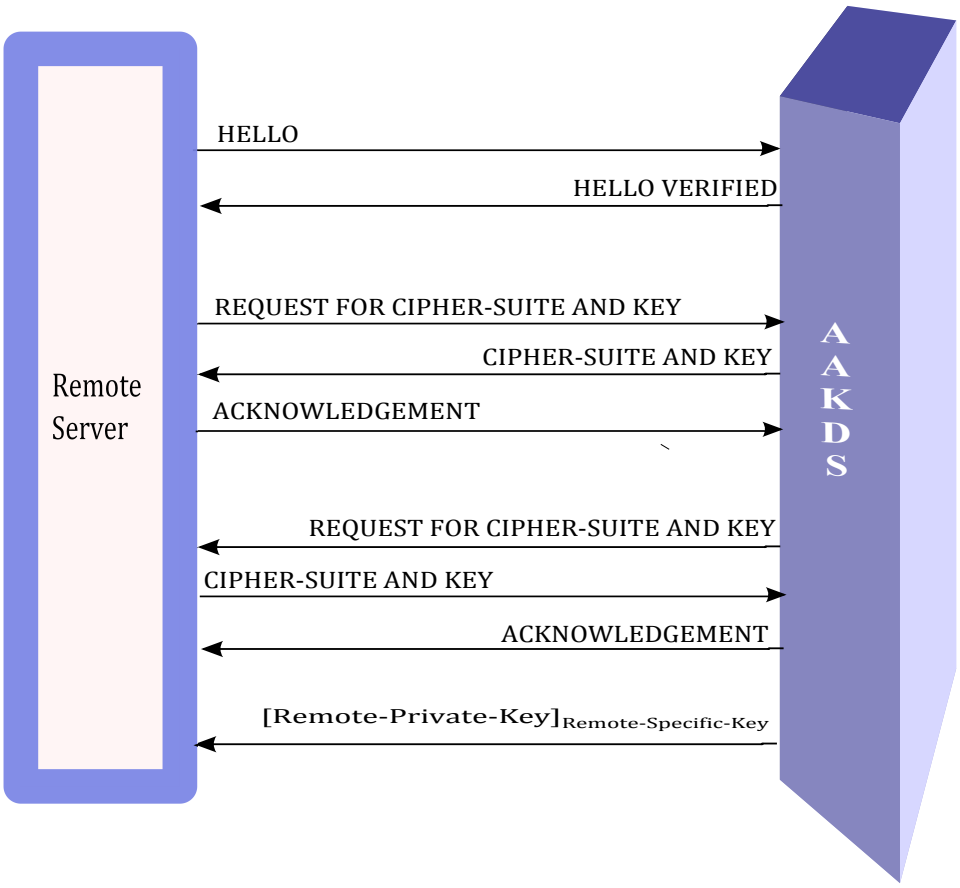


Figure 3.4: Remote Server Registration Process.

Procedure 2 : Remote Server Registration.

1: The Remote Server sends a HELLO message to the AAKDS.

2: The AAKDS sends HELLO VERIFIED message.

3: The Remote Server requests for the cipher-suite and key of the AAKDS.

4: AAKDS sends it's cipher-suite and key and waits for the acknowledgment.

5: AAKDS requests the Remote Server for it's cipher-suite and keys.

6: The Remote Server sends it's cipher-suite and key and waits for the acknowledgment.

3.5.3 Authentication and Authorization of Entities

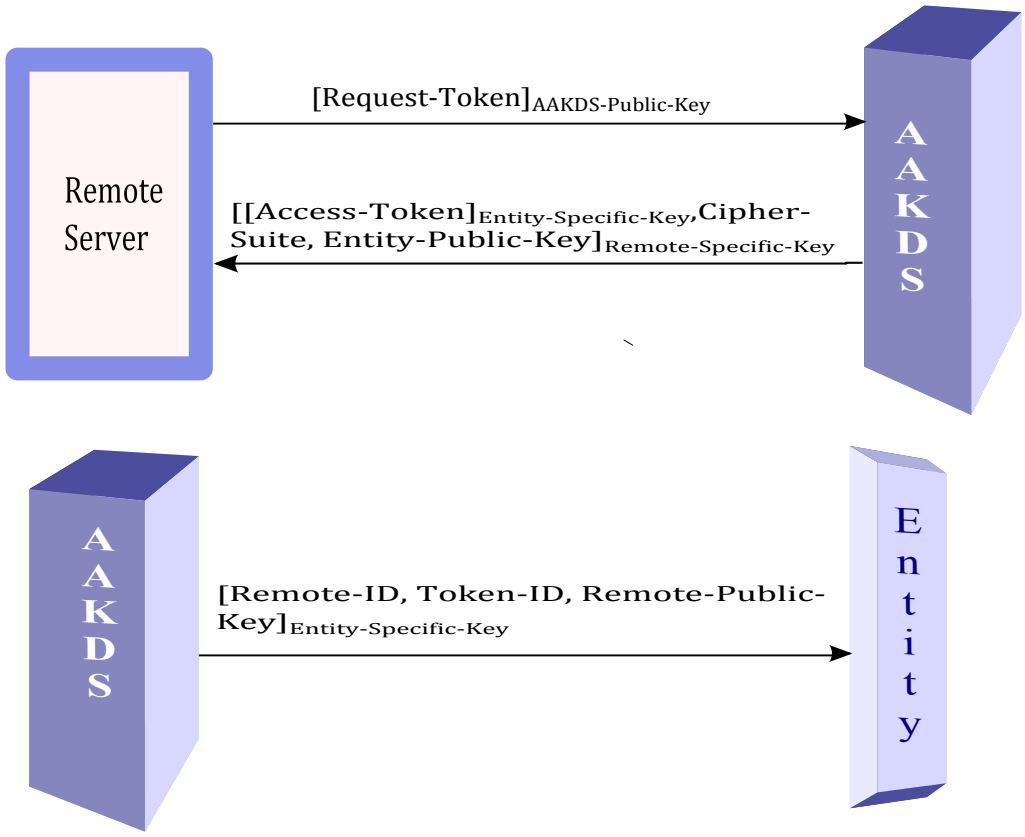
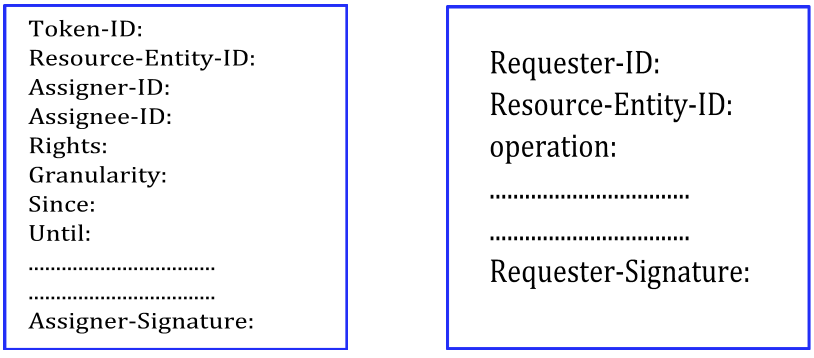


Figure 3.5: Authentication Process.

The authentication and authorization of entities use the following steps shown in Figure 3.5:



(a) Access Token. (b) Request Token.

Figure 3.6: Tokens used for Authentication.

Procedure 3 : Authentication and Authorization.

1: The Remote Server sends a request token, as shown in Figure 3.6, to the AAKDS to access an entity.

2: AAKDS checks the authenticity of the Remote Server and finalize its Authorization.

3: AAKDS then issues an access token, as shown in Figure 3.6, to the Remote Server encrypted with the Entity-specific-key. The access token represents the capability token. It also sends the Entity-public-key.

4: The Remote-ID and the Token-ID is sent to the entity encrypted with the Entity-specific-key.

5: The Entity stores the Remote-ID and the Token-ID for future verification.

3.5.4 Communication

The communication process is shown in Figure 3.7.

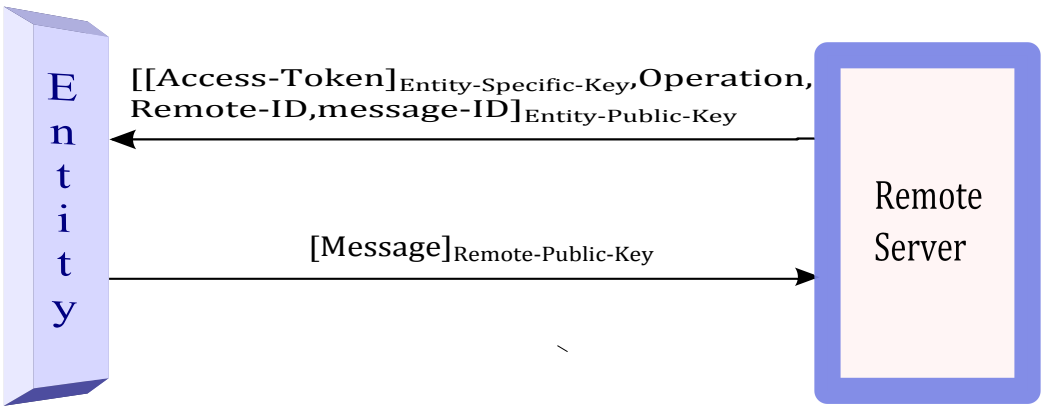


Figure 3.7: Communication Process.

Procedure 4 : Communication.

1: The Remote Server sends a message containing access token,Remote-ID, operation to perform, and a message-ID encrypted with the Entity-public-key.

2: The entity decrypts it and verifies the access token.

3: The entity sends the required response after verifying the authorization.

4: If the Remote Server is not authenticated, then the operation request is rejected.

Two entities also communicate using the same procedure.

3.6 Security Mechanisms

The proposed model takes care of the different security mechanisms to provide the security services: Data Confidentiality, Data Integrity, Authentication, Nonrepudiation and Access Control.

- *Encipherment:* Two techniques are generally used for enciphering: cryptography and steganography. We use cryptography for encipherment which helps to achieve confidentiality.
- *Data Integrity:* The integrity of data is also achieved using cryptography techniques.
- *Digital Signature:* Since public key cryptography is used in our framework, digital signature scheme can be added.
- *Authentication Exchange:* Two entities exchange their identities with the AAKDS, and they use the keys and access token provided by the AAKDS to verify each other.
- *Traffic Padding:* Some bogus data can be added during communication to thwart the traffic analysis.
- *Notarization:* since AAKDS is involved in authentication exchange, one entity can not deny the request or response.
- *Access Control:* A special capability-based Access Token is used which confirms access control.

Chapter 4

Implementation and Results

This chapter provides the details of the implementation of our framework. The cryptographic algorithms are implemented using AVRcryptolib library [25]. The framework is tested using COOJA simulator [6] provided by Contiki OS. The results show the time taken for different flights for each event.

4.1 COOJA

COOJA Simulator is a network simulator provides a Wireless Sensor Network environment for IoT. It is provided by Contiki, which, unlike most simulators also supports real hardware platforms to be emulated. It controls and analyzes a Contiki system via a few functions. The sensors that run Contiki OS can be simulated using Cooja , which is written in java. It simulates networks of sensor nodes of different mote types where each node can be of a different type; differing not only in on-board software, but also in the simulated hardware.

COOJA is flexible in that many parts of the simulator can be easily replaced or extended with additional functionality. Example parts that can be extended, include the simulated radio medium, simulated node hardware, and plug-ins for simulated input/output. A simulated node in COOJA has three basic properties: its data memory, the node type, and its hardware peripherals. The data memory is of very small size analogous to the low -power devices used in IoT. Several nodes share the node type and the properties common to all these nodes are determined. For example, similar type nodes execute a common program code on the common simulated hardware peripherals. And similar type nodes are initialized with the common data memory. However, during execution, nodes' data memories will eventually differ depending upon

the different external inputs. The hardware peripherals include pins and ports for power supply, communication and I/O purpose.

4.2 MSP430 Mote-type

The MSP430 mote-type denotes a 16-bit micro-controller from Texas Instrument (TI). This is designed for low cost and low energy consuming embedded applications, as it is the case in IoT. The basic MSP430 MCUs have an address space of 64 KBytes and can only address memory, including program code, data and peripherals, within this range.

The Z1 mote which is used in simulation has the following properties.

— has a second generation MSP430F2617 low power microcontroller,

— has a CC2420 transceiver, which operates at 2.4GHz with an effective data rate of 250Kbps,

— has abuilt-in digital sensors ready to work,

— has a digital programmable accelerometer (ADXL345),

— has a digital temperature sensor (TMP102),

— does not require additional HW to program,

— provides maximum efficiency and robustness with low energy cost.

4.3 Performance Analysis

The performance is analyzed taking the following considerations: Entities use Symmetric-Key cryptography to communicate with the AAKDS. For the inter-entity communication Asymmetric-Key cryptography is used. The Request-Token and The Access-Token are shown in Figure 4.1.

The following 4 sets of algorithms are used for the implementation of different events and the comparisons are shown below.

- RSA [26] & AES-128 [27]
- RSA [26] & PRESENT-128 [28]

- ECC [29] & AES-128 [27]
- ECC [29] & PRESENT-128 [28]

Token-ID	**: 2byte**
Requesting-Entity-ID	**: 16byte**
Assigner-ID	**: 16byte**
Assignee-ID	**: 16byte**
Rights	**: 16byte**
Since	**: 10byte**
Till	**: 10byte**
Total	**: 86byte**

(a) The Access-Token

Requesting-Entity-ID	**: 16byte**
Resource-Entity-ID	**: 16byte**
Operation	**: 16byte**
Total	**: 48byte**

(b) The Request-Token

Figure 4.1: Token Specifications.

The field values used for our tests are:

Token-ID - unique random number

Requesting-Entity-ID - IPV6 address

Resource-Entity-ID - IPV6 address

Assigner-ID - IPV6 address

Assignee-ID - IPV6 address

Rights - characters

since - characters(dd-mm-yyyy form)

till - characters(dd-mm-yyyy form)

4.3.1 Registration of Entities

Registration of entities event is discussed in the section 3.5.1. The specifications used for entity registration in our test bed is shown in Figure 4.2.

Cipher-Suite	: 4obyte
AAKDS-Public-Key	: 64byte
Total	: 104byte

(a) Flight1 from AAKDS to Entity

Cipher-Suite	: 4obyte
Entity-ID	: 16 byte
Entity-Specific-Key	: 16byte
Total	: 72byte

(b) Flight2 from Entity to AAKDS

Cipher-Suite	: 4obyte
Entity-Private-Key	: 64byte
Total	: 104byte

(c) Flight3 from AAKDS to Entity

Figure 4.2: Flights used for Entity Registration Event.

The comparison of time unit taken for entity registration using different cryptographic algorithm sets are shown in the table 4.1

Table 4.1: Comparison of different algorithm sets for Entity Registration.

Algorithm / Time	time-unit taken for connection	time-unit taken for round-about transmission	Total time-unit taken
RSA & AES	00:02.488	12:48.220	12:50.708
RSA & PRESENT	00:02.488	12:10.977	12:13.465
ECC & AES	00:02.488	10:43.285	10:45.773
ECC & PRESENT	00:02.488	10:15.472	10:17.960

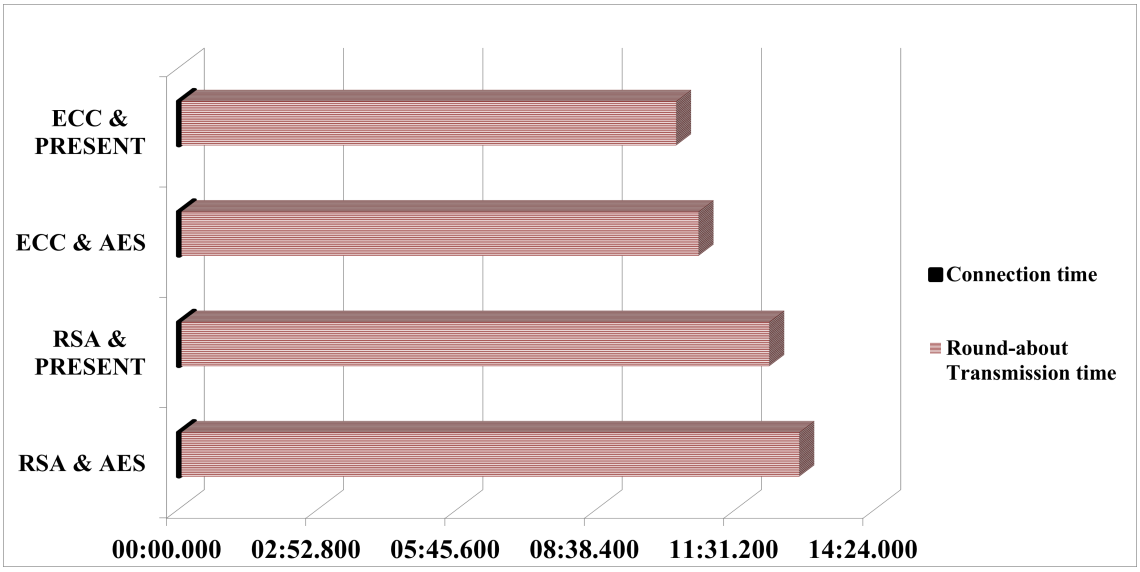


Figure 4.3: Entity Registration Plot.

4.3.2 Authentication Checking

Requesting-Entity-ID	: 16byte
Resource-Entity-ID	: 16byte
Operation	: 16byte
Total	: 48byte

(a) Flight1:request from Entity to AAKDS

Token-ID	: 2byte
Requesting-Entity-ID	: 16byte
Assigner-ID	: 16byte
Assignee-ID	: 16byte
Rights	: 16byte
Since	: 10byte
Till	: 10byte
Resource-Entity-Public-Key	: 64byte
Cipher-Suite	: 40byte
Total	: 190byte

(b) Flight2:respond from AAKDS to Entity

Token-ID	: 2 byte
Requesting-Entity-ID	: 16byte
Requester-Entity-Public-Key	: 64byte
Cipher-Suite	: 40byte
Total	: 122byte

(c) Flight3 from AAKDS to resource Entity

Figure 4.4: Flights used for Authentication Event.

Authorization and Authentication event is discussed in the section 3.5.3. The specifications used for entity registration in our test bed is shown in Figure 4.4.

Table 4.2: Comparison of different algorithm sets for Authentication Process.

Algorithm / Time	time-unit taken for connection	time-unit taken for round-about transmission	Total time-unit taken
RSA & AES	00:02.488	11:10.376	11:12.864
RSA & PRESENT	00:02.488	10:34.048	10:36.536
ECC & AES	00:02.488	09:28.316	09:30.804
ECC & PRESENT	00:02.488	09:09.962	09:12.450

The comparison of time unit taken for authentication process, using different cryptographic algorithm sets, are shown in the table 4.2.

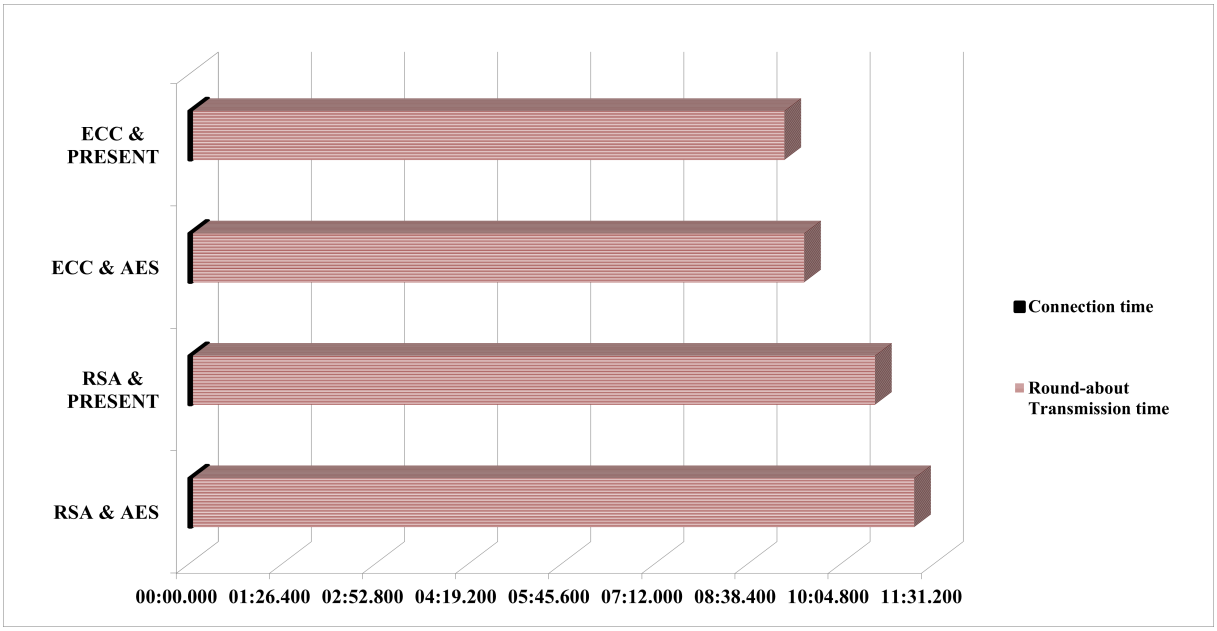


Figure 4.5: Entity Authentication Plot.

4.3.3 Communication

Communication between the entities event is discussed in the section 3.5.4. The specifications used in our test bed for this event is shown in Figure 4.6.

Access-Token	**: 86byte**
Requesting-Entity-ID	**: 16byte**
Operation	**: 16byte**
Nonce	**: 2byte**
Total	**: 120byte**

(a) Flight1:request from Entity to AAKDS

Requesting-Entity-ID	**: 16byte**
Resource-Entity-ID	**: 16byte**
message	**: 16byte**
Total	**: 48byte**

(b) Flight2:respond from AAKDS to Entity

Figure 4.6: Flights sent to access a service.

The comparison of time unit taken for requesting a service and getting the respond using different cryptographic algorithm sets are shown in the table 4.3

Table 4.3: Comparison of different algorithm-sets for Communication Event.

Algorithm / Time	time-unit taken for connection	time-unit taken for round-about transmission	Total time-unit taken
RSA & AES	00:02.488	11:48.583	11:51.071
RSA & PRESENT	00:02.488	11:16.604	11:19.092
ECC & AES	00:02.488	09:58.294	10:00.782
ECC & PRESENT	00:02.488	09:34.296	09:36.784

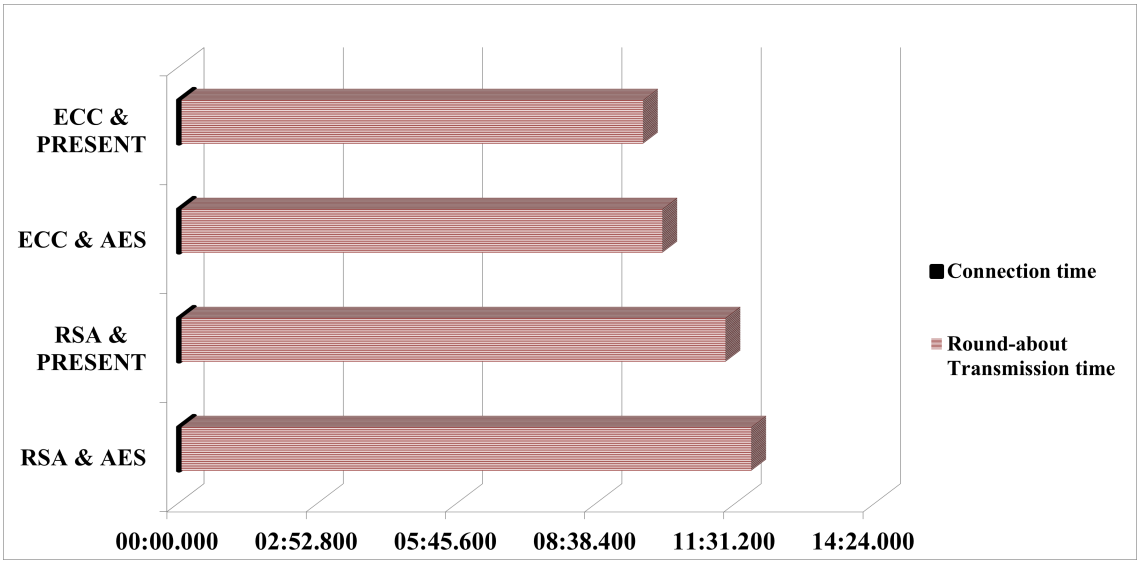


Figure 4.7: Communication Plot.

4.4 Chapter Summary

From the above tables and graphs it is identified that the connection time is almost negligible as compared to the computation time and communication time. The implementation of PRESENT algorithm for the encryption and decryption of the message between the Entity and the AAKDS, and , ECC algorithm for the encryption and decryption of the message used for inter-Entity communication, takes less time for each event. Finally it is realizable that, the capability based authentication and access control can be implemented in constrained network and it gives satisfactory performance.

Chapter 5

Hardware Emulation

This chapter provides the details of the hardware implementation of our framework. We prepare a testbed where Arduino Uno micro-controller is used as the entity to provide the services. In the Arduino board the cryptographic algorithms are implemented using arduino IDE 1.6.3 [30] and AVRcryptolib library [25]. We analyze the memory space taken to implement the cryptographic algorithms..

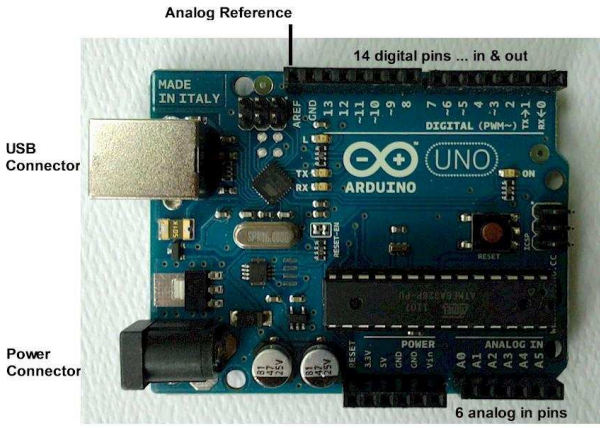
5.1 Hardware Requirement

For our hardware implementation we use the following devices:

- 2 Arduino uno R3 micro-controller boards,
- 2 ESP8266 wifi modules,
- 2 bread-boards,
- 1 LED,
- jumper-wires.

The Arduino Uno R3 microcontroller board and it's specifications are shown in Figure 5.1

The ESP8266 wifi module with it's pin specification and it's connection specification with Arduino uno R3 is shown in Figure 5.2

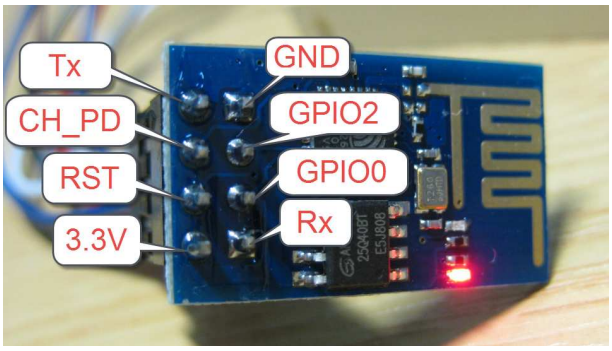


(a) arduino uno R3

Micro-controller	:	ATmega328
Flash Memory	:	32Kb
SRAM	:	2048b
operating voltage	:	5v
Input voltage	:	7-12v
Digital I/O pins	:	14(6PWM output)
Analog input pins	:	6
DC current per I/O pin	:	40mA
DC current per 3.3v pin	:	50mA
clock speed	:	16 mHz

(b) arduino uno R3 specfications

Figure 5.1: Arduino Uno R3 micro-controller Board.



(a) ESP8266 wifi module

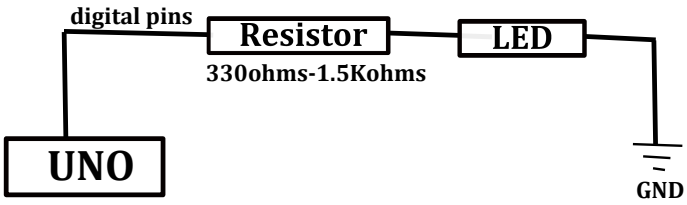
Instruction RAM	:	64Kb
Data RAM	:	96Kb
ROM	:	64 Kb
operating voltage	:	3.3v
Input voltage	:	7-12v
Analog input pins	:	8
clock speed	:	80 mHz

(b) ESP8266 specfications

Figure 5.2: ESP8226 micro-controller Board.

5.2 Emulation and Result Analysis

We create a testbed as below. The connection specifications are shown in Figure 5.3.



(a) arduino uno R3 and LED

Arduino pins		ESP8266 pins
Tx	<---------->	Rx
Rx	<---------->	Tx
3.3v	<---------->	Vcc
GND	<---------->	GND
3.3v	<---------->	CH_POD

(b) arduino uno R3 and ESP8266

Figure 5.3: Connection Specifications.

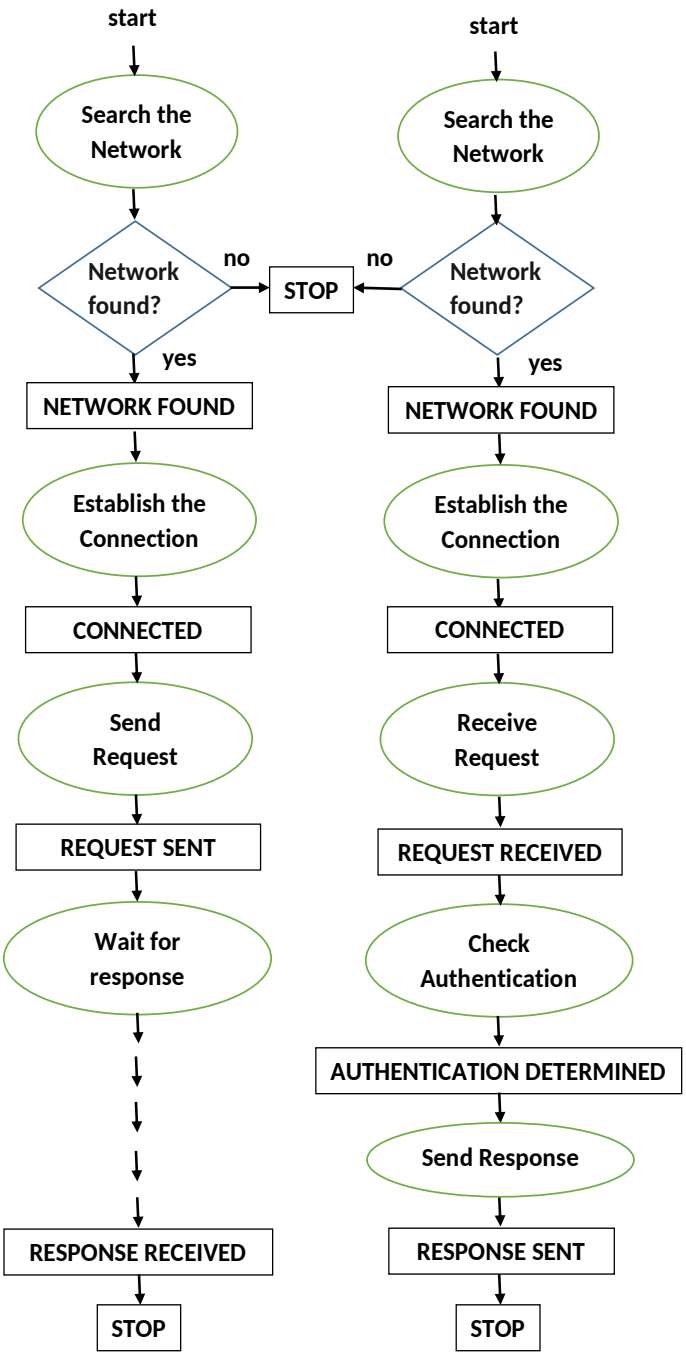


Figure 5.4: State Transition Diagram for the Requesting Entity (left) and the Resource Entity(right).

- First arduino microcontroller is connected with an LED. It provides wifi service using ESP8266 wifi module and serves as the resource entity. It provides 3 services: 1) only Read i.e., the requesting entity can only query whether the LED is ON or OFF, 2) Both Read and Write i.e., the requesting entity can inquire the

status of the LED and also toggle it, 3) The requesting entity can not access the status.

- The second arduino microcontroller is also connected to ESP8266 wifi module to communicate with the first one. It serves as the requesting entity.
- The key values and the cipher-suites are already set.

The state transition diagram for the requesting entity and resource entity are shown in Figure 5.4.

The RAM size used by implementing the different cryptographic algorithm sets mentioned in section 4.3 are shown in the table 5.1.

Table 5.1: Comparison of RAM capacity used.

Algorithm / Time	RAM size occupied
RSA & AES	1007 byte
RSA & PRESENT	961 byte
ECC & AES	1198 byte
ECC & PRESENT	1152 byte

5.3 Chapter Summary

From the hardware implementation of our framework and quick analysis of the RAM capacity used by different algorithm sets given in the table 5.1, we find that ECC and PRESENT combination requires more RAM capacity. But we still suggest this for better performance, since ECC gives same security as RSA with smaller key size according to NIST recommendation of key size [31].

Chapter 6

Conclusions and Future Work

To summarize, the work gives a framework that can be used by organizations to protect and manage their information in the environment of IOT. As part of the work the different light weight cryptographic algorithms are tested in the proposed framework. Under the test environment, the developed prototype has delivered promising performance results. Hence the proposed framework might be considered for use in real world scenarios.

6.1 Suggestions for Improvement

The following key points are suggested as the extension of our work

- Our evaluation focuses on the constrained networks. However, in the IOT we need to consider the interaction of the constrained devices with the traditional Internet i.e., a Remote Sever may be given the access control as proposed in our framework. COOJA is not suitable for this simulation. Still it can be tested by developing a web-server application which perform the task of Remote Server. A cryptography library JScrypto, written in Jscript can be used for cryptographic implementations.

- Our work may not meet the requirements for each and every use case of the vast IOT domain, like in the transmission of real time data stream.

- In a IOT domain an entity can work both as a client and a server. So, a novel alternate authentication approach is needed for an entity to be able to delegate authentication to another entity.

- For performance benefits, newer cryptographic algorithms can be explored.
- When entities communicate directly a port has to be opened in the router, which creates a play field for the hackers. Hence an alternate model can be explored where a cloud sever can be used as an intermediary.

Bibliography

[1] K. Ashton, "That internet of things thing," *RFiD Journal*, vol. 22, no. 7, pp. 97–114, 2009.

[2] Friedemann Mattern. Hundert Jahre Zukunft,Visionen zum, Available: http://link.springer.com/chapter/10.1007%2F978-3-540-71455-2_18.

[3] T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246, IETF," August 2008, Available: http://tools.ietf.org/html/rfc5246.

[4] S. Frankel, S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap. RFC 6071, IETF," February 2011, Available: http://tools.ietf.org/html/rfc6071.

[5] E. Rescorla, N. Modadugu, "Datagram Transport Layer Security Version 1.2. RFC 6347, IETF," January 2012, Available: http://tools.ietf.org/html/rfc6347.

[6] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. IEEE, 2006, pp. 641–648.

[7] Z. Shelby, S. Chakrabarti, E. Nordmark, C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). RFC 6775, IETF," November 2012, Available: http://tools.ietf.org/html/rfc6775.

[8] E. Callaway, P. Gorday, L. Hester, J. A. Gutierrez, M. Naeve, B. Heile, and V. Bahl, "Home networking with ieee 802. 15. 4: a developing standard for low-rate wireless personal area networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 70–77, 2002.

[9] N. Venkatesh, "Ultra-low power 802.11 n wi-fi–wireless connectivity for the internet of things.," *Low-Power Wireless as White Paper, Last visited on*, vol. 16, p. 2013, 2010.

[10] S. Bluetooth, "The bluetooth core specification, v4. 0," *Bluetooth SIG: San Jose, CA, USA*, 2010.

[11] N. Kushalnagar, G. Montenegro, C. Schumacher, "IPv6 over Low-Power WirelessPersonal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, IETF," August 2007, Available: http://tools.ietf.org/html/rfc4919.

[12] G. Montenegro, N. Kushalnagar, J. Hui, D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, IETF," September 2007, Available: http://tools.ietf.org/html/rfc4944.

[13] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, IETF," May 2008, Available: http://tools.ietf.org/html/rfc5280.

[14] T. M. Heer, *Direct end-to-middle authentication in cooperative networks*. Universitätsbibliothek, 2011.

[15] M. Campagna, D. Stebila, "ECMQV ECQV Cipher Suites for Transport Layer Security (TLS). draft-campagna-tls-ecmqv-ecqv-01, IETF," January 2010, Available: http://tools.ietf.org/html/draft-campagna-tls-ecmqv-ecqv-01.

[16] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*. IEEE, 2004, pp. 455–462.

[17] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks," *Wireless networks*, vol. 8, no. 5, pp. 521–534, 2002.

[18] O. G. Morchon and H. Baldus, "Efficient distributed security for wireless medical sensor networks," in *Intelligent Sensors, Sensor Networks and Information Processing, 2008. ISSNIP 2008. International Conference on*. IEEE, 2008, pp. 249–254.

[19] T. Freeman, A. Malpani, D. Cooper, and R. Housley, "Server-based certificate validation protocol (scvp)," 2007.

[20] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, and M. Rossi, "Secure communication for smart iot objects: Protocol stacks, use cases and practical examples," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*. IEEE, 2012, pp. 1–7.

[21] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, S. Guna, G. P. Jesi, R. Lo Cigno, L. Mottola, A. L. Murphy *et al.*, "Is there light at the ends of the tunnel? wireless sensor networks for adaptive lighting in road tunnels," in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*. IEEE, 2011, pp. 187–198.

[22] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle *et al.*, "Permadaq: A scientific instrument for precision sensing and data recovery in environmental extremes," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*. IEEE Computer Society, 2009, pp. 265–276.

[23] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the ip-based internet of things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, 2011.

[24] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the internet of things," *Mathematical and Computer Modelling*, vol. 58, no. 5, pp. 1189–1205, 2013.

[25] D. Otte, "Avr–crypto–lib.–2009," *Online: http://www. das–labor. org/wiki/AVR–Crypto–Lib/en*, 2009.

[26] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[27] P. Chown, "Advanced encryption standard (aes) ciphersuites for transport layer security (tls)," 2002.

[28] A. Y. Poschmann, "Lightweight cryptography: cryptographic engineering for a pervasive world," in *Ph. D. Thesis*. Citeseer, 2009.

[29] B. A. Forouzan, *Cryptography & Network Security*. McGraw-Hill, Inc., 2007.

[30] Arduino IDE software, Available: http://www.arduino.cc/en/main/Software.

[31] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz, "Comparing elliptic curve cryptography and rsa on 8-bit cpus," in *Cryptographic Hardware and Embedded Systems-CHES 2004*. Springer, 2004, pp. 119–132.